Exporting/Importing an Informix Database

Should you wish to move an Informix database from one place to another, this procedure should work.

On the source server, set the Informix path and database directory to the place where Informix was installed and the place where the database lives:

```
        PATH=/opt/informix/bin:$PATH
        DBPATH=/home/Informix/chunks
        export PATH DBPATH
```

Running as the "informix" user, export the database to flat files:

```
        su informix
        dbexport -o /export/dir dbname
```

This will create a bunch of export files in the path /export/dir/dbname.exp. If you want to FTP this to another machine, tar it up:

```
        cd /export/dir
        tar -cvf dbname.exp.tar dbname.exp
```

or

```
        cd /export/dir
        tar -cvf - dbname.exp | gzip -c >dbname.exp.tar.gz
```

FTP the tar file to other machine and then untar it:

```
        cd /import/dir
        tar -xvf dbname.exp.tar
```

or

```
        cd /import/dir
        tar -xvzf dbname.exp.tar.gz
```

Set up the environment for importing:

```
        PATH=/opt/informix/bin:$PATH
        DBPATH=/import/dir
        INFORMIXDIR=/opt/informix
        INFORMIXSERVER=dev
        export PATH DBPATH INFORMIXDIR INFORMIXSERVER
```

Run the import as the informix user:

```
       su informix
       dbimport -i /import/dir [-l /log/file/name [ansi]]
  dbname
```

If your database needs transaction logging (you can find out by running the SQL command "select dirpath from systables where tabid = 0" in the original table), you'll need to create the logfile when you do the import, using the "-l" parameter and possibly the "ansi" parameter. If you don't do it at this time, you are screwed because Informix is so brain-dead as to not allow you to add the log after the fact.

Also, be sure to use an absolute path name for the logfile since the "-l" option appears not to work for Informix under Linux when relative path names are used. All that happens is a "database not found" error. Any "-l" option with a relative path appears to cause the problem, even when it is in the current directory and/or the directory where the database will live.

One word of caution, though. Full-blown logging, transactions and rollback are not something most applications are prepared to deal with so, if you use the "ansi" parameter, you are probably screwed too. It doesn't seem to work very well with Informix so it might be a good idea not to use it for anything but a special database, where you really know what you're doing.

The permissions on any imported databases will probably be set wrong. You can either give each database directory and the files within it all permissions (at a minimum, the user permissions are not set so you should at least add them) or you can create a group that each person who will access the database can belong to and then make the group of each database this group.

To set general permissions (admittedly a security hole), do something like this:

```
       chmod ugo=rwx /home/coll/collprod/coll.dbs
       chmod ugo=rwx /home/coll/collprod/coll.dbs/*
```

To add each user of the database to a group that can access the database, first create the group (as super user), in this fashion:

```
       /usr/sbin/groupadd collusers
```

Add each of the users who must access the database to the group (once again as super user), something like this:

```
       gpasswd -a coll collusers
```

Finally, set the group of the database directory and all its files to the group just created, for example:

```
       chgrp collusers /home/coll/collprod/coll.dbs
       chgrp collusers /home/coll/collprod/coll.dbs/*
```

We now include a couple of scripts that we use to export and import a production database from the production server to the hotbackup server, on a nightly basis. These scripts should be fairly self-explanatory:

DBExport:

```
     #!/bin/sh
     #
     # Shell script to export the production database and
create a tar file.
     #
     # This script must be run as the informix user.  Either
logon as informix or
     # do the following as root:
     #
     #      su -c /bin/path/DBExport informix
     #
     #
     # Clean up the export directory, if there's any left-
over junk.
     #
     rm -rf /backup/export/*
     #
     # Set up the environment variables needed by Informix.
     #
     INFORMIXDIR=/opt/informix
     INFORMIXSERVER=dev
     DBPATH=/home/informix/chunks
     #
     # Set the path to include the Informix binary directory
at the front, if it
     # isn't already there.
     #
     echo $PATH | grep -q "^$INFORMIXDIR/bin"
     DirVal=$?
     if [ $DirVal != 0 ]; then
         PATH=$INFORMIXDIR/bin:$PATH
     fi
     export INFORMIXDIR INFORMIXSERVER DBPATH PATH
     #
     # Change to the production directory.
     #
     cd /home/informix/chunks
     #
     # Export the production database to the export
directory.  This will create
     # /backup/export/prod.exp.
     #
     dbexport -o /backup/export prod 2>&1 >/dev/null
     #
     # Tar and gzip the exported database so that FTP can
copy it.
     #
     cd /backup/export
     tar -cf - prod.exp | gzip
>/backup/export/prod.exp.tar.gz
     #
     # We don't need the export directory any longer so
we'll delete it.  All we
     # need is the gzipped tar file.
     #
     rm -rf /backup/export/prod.exp
     #
     # FTP the exported database to its destination.
     #
     echo -e "user prod shhh\\nbin\\nput
/backup/export/prod.exp.tar.gz /home/prod/prod.exp.tar.gz" |
\
         ftp -n 10.100.0.1 2>&1 >/dev/null
```

DBImport:

```
     #!/bin/sh
     #
     # Shell script to import the production database from a
tar file.
     #
     # This script must be run as the informix user.  Either
logon as informix
     # or do the following as root:
     #
     #      su -c /bin/path/DBImport informix
     #
     #
     # Check to see if there's anything worth doing.
     #
     if test /home/prod/lastexport -nt
/home/prod/prod.exp.tar.gz; then
         echo The database is right up to date with the last
export.
         exit 0
     fi
     #
     # Also, check to see if the export crapped out or looks
OK.
     #
     TarFile=`find /home/prod -name prod.exp.tar.gz -follow
-size +1024k -print`
     if test x"$TarFile" == x; then
```

```
         echo Looks like the export of the production
database failed.
         exit 1
     fi
     #
     # Set up the environment variables needed by Informix.
     #
     INFORMIXDIR=/opt/informix
     INFORMIXSERVER=dev
     DBPATH=/home/prod/prod
     #
     # Set the path to include the Informix binary directory
at the front, if it
     # isn't already there.
     #
     echo $PATH | grep -q "^$INFORMIXDIR/bin"
     DirVal=$?
     if [ $DirVal != 0 ]; then
         PATH=$INFORMIXDIR/bin:$PATH
     fi
     export INFORMIXDIR INFORMIXSERVER DBPATH PATH
     #
     # Change to the production directory.
     #
     cd $DBPATH
     #
     # Just in case, get rid of the old export directory.
     #
     rm -rf  $DBPATH/prod.exp
     #
     # Extract the export directory from the tar file.  The
export
     # (DBExport) thoughtfully copied it where we could find
it (in
     # /home/prod/prod.exp.tar.gz).
     #
     echo Extracting the exported database.
     tar -xzf /home/prod/prod.exp.tar.gz
     RetVal=$?
     if test $RetVal != 0; then
         echo Extract of production database from tar file
failed.
         exit 2
     fi
     #
     # Now that we have something good to install on this
machine, get rid of
     # the old database.
     #
     echo Making a back-em-up of the current database.
     rm -rf $DBPATH/prod.dbs.bak
     mv $DBPATH/prod.dbs $DBPATH/prod.dbs.bak
     rm -f $DBPATH/prod.log
     #
     # Import the production database from the export
directory (which we built,
     # above).
     #
     echo Loading the new database.
     dbimport -i $DBPATH -l $DBPATH/prod.log prod 2>&1
>/dev/null
     RetVal=$?
     if test $RetVal != 0; then
         echo Load of the new production database failed.
         rm -rf $DBPATH/prod.dbs
         mv $DBPATH/prod.dbs.bak $DBPATH/prod.dbs
         exit 3
     fi
     #
     # We don't need the export directory any longer so
we'll delete it.  We'll
     # keep the tar file for yucks.  We also don't need the
data in the logfile.
     #
     echo Cleaning up after load of new database.
     rm -rf  $DBPATH/prod.exp
     cat /dev/null >$DBPATH/prod.log
     #
     # Save the old database for posterity.
     #
     SaveDate=`date +%y%b%d`
     mv $DBPATH/prod.dbs.bak $DBPATH/prod-$SaveDate.dbs
     echo Old database saved as $DBPATH/prod-$SaveDate.dbs
     #
     # Indicate we're done with these tar files.
     #
     touch /home/prod/lastexport
     #
     # Since we're doing target practice here, we'll make a
copy of the database
     # for quick restores.
     #
     echo Copying new database to quick restore database
     rm -f $DBPATH/quickrest.dbs/
     cp $DBPATH/prod.dbs/ $DBPATH/quickrest.dbs
```